UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO.	: 7,039,919 B1
APPLICATION NO.	: 09/196836
DATED	: May 2, 2006
INVENTOR(S)	: Hunt

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On the Title pg on page 2, Item -56-, under "Other Publications", line 6, delete "Effifient" and insert -- Efficient --, therefor.

On the Title pg on page 2, Item -56-, under "Other Publications", line 23, delete "Hunt,"Inter" and insert -- Hunt, "Inter --, therefor.

On the Title pg on page 3, Item -56-, under "Other Publications", line 36, delete "78-88" and insert -- 79-88 --, therefor.

In column 3, line 34, delete "In" and insert -- in --, therefor.

In column 3, lines 42–67 and column 4, lines 1–10, delete
"To profile an application, an ADPS may measure communication between units of the application. To do so, the ADPS needs to identify units of the application. In an object-oriented framework, an object may not readily present an identity for the ADPS to use during profiling. For example, a COM object presents identifiable interfaces and belongs to an identifiable class, but presents no identity that is readily useable by an ADPS, or by any type of instrumentation system for that matter. Neither ICOPS, CAGES, nor IDAP provides a mechanism for determining which units of an application program expose which interfaces in an environment in which units lack readily available **identities. An** APDS must recognize and treat location constraints on units. In ICOPS, CAGES, and IDAP, a programmer manually indicates location constraints for units of an application. There are numerous techniques for manually recognizing location constraints. A programmer can insert into application source code a call to an ADPS to indicate a location constraint for a unit. Application units can be "marked" in code so that static analysis of the code detects units marked as location constrained. A programmer can provide to an ADPS a list of units that are location constrained. If a programmer is responsible for directly distributing units, the programmer can recognize and handle location constraints when manually implementing the distribution. While manual recognition of location constraints is acceptable in applications with few units, the task of manual detection dramatically increases in difficulty and complexity as the number of units increases. Moreover, when application units change, location constraints may change too, requiring repetition of the tedious manual process. If a programmer lacks intimate knowledge of the units of an application, the task is further complicated. Nevertheless, neither ICOPS, CAGES, nor IDAP provides a mechanism for automatically detecting per-unit or pair-wise location constraints." and
insert -- To profile an application, an ADPS may measure communication between units of the application. To do so, the ADPS needs to identify units of the application. In an object-oriented framework, an object may not readily present an identity for the ADPS to use during profiling. For example, a COM object presents identifiable interfaces and belongs to an identifiable class, but presents no identity that is readily useable by an ADPS, or by any type of instrumentation system for that matter. Neither ICOPS, CAGES, nor IDAP provides a

mechanism for determining which units of an application program expose which interfaces in an environment in which units lack readily available **identities.**

An APDS must recognize and treat location constraints on units. In ICOPS, CAGES, and IDAP, a programmer manually indicates location constraints for units of an application. There are numerous techniques for manually recognizing location constraints. A programmer can insert into application source code a call to an ADPS to indicate a location constraint for a unit. Application units can be "marked" in code so that static analysis of the code detects units marked as location constrained. A programmer can provide to an ADPS a list of units that are location constrained. If a programmer is responsible for directly distributing units, the programmer can recognize and handle location constraints when manually implementing the distribution. While manual recognition of location constraints is acceptable in applications with few units, the task of manual detection dramatically increases in difficulty and complexity as the number of units increases. Moreover, when application units change, location constraints may change too, requiring repetition of the tedious manual process. If a programmer lacks intimate knowledge of the units of an application, the task is further complicated. Nevertheless, neither ICOPS, CAGES, nor IDAP provides a mechanism for automatically detecting per-unit or pair-wise location constraints. --, therefor.

In column 15, line 39, delete "(DC**B** RPC)" and insert -- (DCE RPC) --, therefor.

In column 25, line 24, delete "256**6**" and insert -- 256 --, therefor.

In column 45, line 65, delete "**w**in32" and insert -- **W**in32 --, therefor.

In column 47, lines 10–11, delete "comm**m**on" and insert -- common --, therefor.

In column 47, line 30, after "linking" insert -- **.** --.

In column 48, line 7, delete "**in to**" and insert -- **into** --, therefor.

In column 53, line 43, delete "Javascript" and insert -- JavaScript --, therefor.

In column 58, line 14, in Claim 11, delete "**%** rapped," and insert -- **w**rapped, --, therefor.

Signed and Sealed this

Twentieth Day of April, 2010

David J. Kappos

David J. Kappos
Director of the United States Patent and Trademark Office